Analisis dan Implementasi Algoritma Rijndael (AES) dan Kriptografi RSA pada Pengamanan File

Rinmar Siringoringo

Universitas Imelda Medan, Jl. Bilal Ujung No.24,52, Pulo Brayan Darat I, Kec. Medan, Sumatera Utara, Indonesia

ARTICLE INFORMATION

Received: Februari, 20, 2020
Revised: Maret, 6, 2020
Available online: April,9 2020

KEYWORDS

Algoritma Rijndael, Keamanan File, AES , RSA

CORRESPONDENCE

Phone: +62 (0751) 12345678

E-mail: rimmarsiringoringo74@gmail.com

A B S T R A K

Salah satu cara untuk meningkatkan keamanan terhadap data atau file yaitu dengan menggunakan metode kriptografi. Pada kriptografi modern, algoritma yang digunakan tidak dirahasiakan sebab setiap kali algoritma diketahui lawan, maka kriptografer (cryptographer) harus membuat algoritma baru, dengan demikian cukup kuncinya yang harus dirahasiakan dan benar-benar dijaga keamanannya. Algoritma Rijndael didesain untuk mengganti algoritma DES yang sudah cukup lama. Algoritma ini menggunakan kunci 128-bit, 192-bit atau 256 bit. Sistem yang dibangun mampu mengenkripsi dan dekripsi file yang dipilih saja. Panjang kunci algoritma Rijndael pada sistem ini hanya 128-bit. Hasil penelitian ini diperoleh sistem enkripsi dan dekripsi terhadap plainteks dan kunci simetris (sessionkey) dengan kombinasi algoritma Rijndael dan RSA. Hasil enkripsi plainteks pada sistem yang dibangun berupa kode karakter, sedangkan untuk enkripsi sessionkey berupa kode number.

PENDAHULUAN

Perkembangan teknologi komunikasi yang semakin pesat saat ini membuat segala urusan sehari-hari manusia tidak terlepas dari penggunaan perangkat teknologi dalam menunjang berbagai aktivitas agar lebih mudah dan cepat. Penggunaan perangkat teknologi saat ini boleh dikatakan wajib mengingat kecepatan waktu merupakan tujuan utama baik dalam hal berkomunikasi atau berbagi informasi mulai dari yang sifatnya rahasia (private) maupun yang sifatnya umum *(public)*[1], [2]. Namun seiring dengan kemajuan tersebut tingkat kejahatan di era teknologi juga semakin meningkat dan berkembang, sehingga sangat perlu diperhatikan segi keamanannya pada saat melakukan komunikasi atau hal-hal lain terutama yang bersifat rahasia.

Salah satu cara untuk meningkatkan keamanan terhadap data atau file yaitu dengan menggunakan metode kriptografi. Pada kriptografi modern, algoritma yang digunakan tidak dirahasiakan sebab setiap kali algoritma diketahui lawan, maka kriptografer (cryptographer) harus membuat algoritma baru, dengan demikian cukup kuncinya yang harus dirahasiakan dan benar-benar dijaga keamanannya. Berdasarkan jenis kunci yang digunakan, kriptografi terbagi atas dua metode, yaitu kriptografi kunci simetris dan kriptografi kunci asimetris. Perbedaan dari kedua kriptografi ini terletak pada penggunaan kunci [3]. Untuk kriptografi simetris menggunakan kunci yang sama pada saat melakukan enkripsi dan dekripsi. Oleh sebab itu harus benar-benar dijaga kerahasian kunci tersebut, namun berbeda halnya dengan kriptografi asimetris, kunci pada saat enkripsi berbeda dengan kunci yang digunakan pada saat melakukan dekripsi, hal ini menjadi salah satu faktor kriptografi asimetris lebih aman dibandingkan dengan kriptografi simetris [4].

Berdasarkan uraian di atas, sistem kriptografi secara mutlak ditentukan oleh keamanan kunci yang digunakan. Selain panjangnya kunci, proses pada saat pertukaran kunci harus juga diperhatikan agar kunci tersebut tetap aman. Algoritma kriptografi kunci publik atau sering disebut dengan algoritma kunci asimetris terkadang tidak pernah berdiri sendiri, algoritma kunci publik juga membutuhkan algoritma kunci simetris. Dalam hal ini biasanya algoritma kunci simetris tersebut digunakan untuk enkripsi dan dekripsi plainteks karena dari segi kecepatan waktu algoritma simetris cukup menguntungkan, sedangkan algoritma kunci publik berperan untuk mengenkripsikan kunci dari kunci simetris tersebut agar lebih aman pada saat pendistribusian kunci dan pesan.

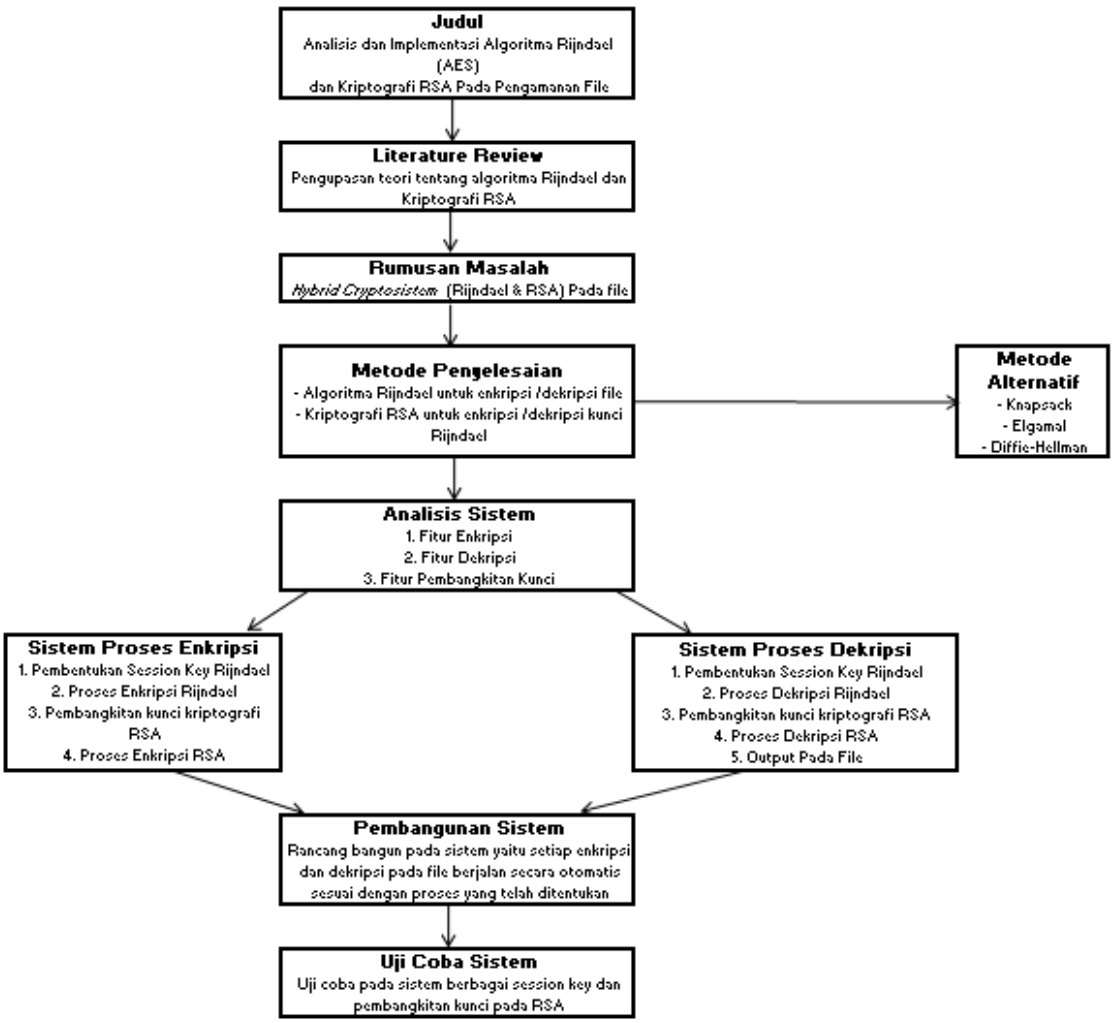
Berdasarkan permasalahan itu penulis berencana untuk mengombinasikan kedua sistem kriptografi tersebut. Untuk kriptografi simetris penulis memilih algoritma kriptografi Rijndael yang merupakan pemenang dalam kompetisi yang dilakukan oleh NIST (National Institute of standard and Tekhnology) pada 2 Oktober 2000 untuk menggantikan algoritma DES (Data Encryption Standard) yang dirasa sudah tidak aman lagi dalam penggunaannya. Algoritma Rijndael didesain untuk mengganti algoritma DES yang sudah cukup lama. Algoritma ini menggunakan kunci 128-bit, 192-bit atau 256 bit [1]. Algoritma ini dibuat oleh dua orang periset yaitu Joan Daemen dan Vincent Rijmen. Rijndael telah dipilih oleh NIST (National Institute of Standards and Technology) sebagai proposal yang paling cocok dengan kriteria keamanan, efisiensi dalam implementasi, sifat yang berubah-ubah dan kesederhanaan.
Terkadang istilah AES dan Rijndael digunakan tertukar-tukar, tetapi keduanya berbeda. AES secara luas diperkirakan untuk menjadi standard utama dalam mengenkripsi semua bentuk data elektronik termasuk data yang digunakan dalam aplikasi komersial seperti perbankan dan transaksi finansial, telekomunikasi, dan informasi privat. Algoritma memiliki beberapa kelebihan antara lain relatif

cepat dalam perangkat lunak dan perangkat keras, mudah untuk diimplementasikan dan hanya menggunakan memori yang kecil [5], [6]. Sedangkan untuk sistem pertukaran kuncinya menggunakan kriptografi asimetris yaitu RSA, algoritma ini pertama kali dipublikasikan di tahun 1977 oleh Ron Rivest, Adi Shamir, dan Leonard Adleman dari Massachusetts Institute of Technology (MIT). Berdasarkan uraian pada latar belakang di atas, maka dapat diambil suatu permasalahan bagaimana cara mengimplementasikan dua buah algoritma yang berbeda yaitu Rijndael dan RSA untuk melakukan proses enkripsi dan dekripsi. Sistem ini lebih dikenal dengan Hybrid Cryptosistem [7].

METODE PENELITIAN

2.1. Kerangka Pemikiran

Kerangka pemikiran adalah narasi (uraian) atau pernyataan (proposisi) tentang kerangka konsep pemecahan masalah yang telah diidentifikasi atau dirumuskan. Kerangka berpikir atau kerangka pemikiran dalam sebuah penelitian kuantitatif, sangat menentukan kejelasan dan validitas proses penelitian secara keseluruhan. Melalui uraian dalam kerangka berpikir, peneliti dapat menjelaskan secara komprehensif variabel-variabel apa saja yang diteliti dan dari teori apa variabel-variabel itu diturunkan, serta mengapa variabel-variabel itu saja yang diteliti. Uraian dalam kerangka berpikir harus mampu menjelaskan dan menegaskan secara komprehensif asal-usul variabel yang diteliti, sehingga variabel-variabel yang tercatum di dalam rumusan masalah dan identifikasi masalah semakin jelas asal-usulnya. Berikut ini adalah bagan dari kerangka pemikiran yang telah disusun oleh penulis.



Gambar 1. Kerangka Pemikiran

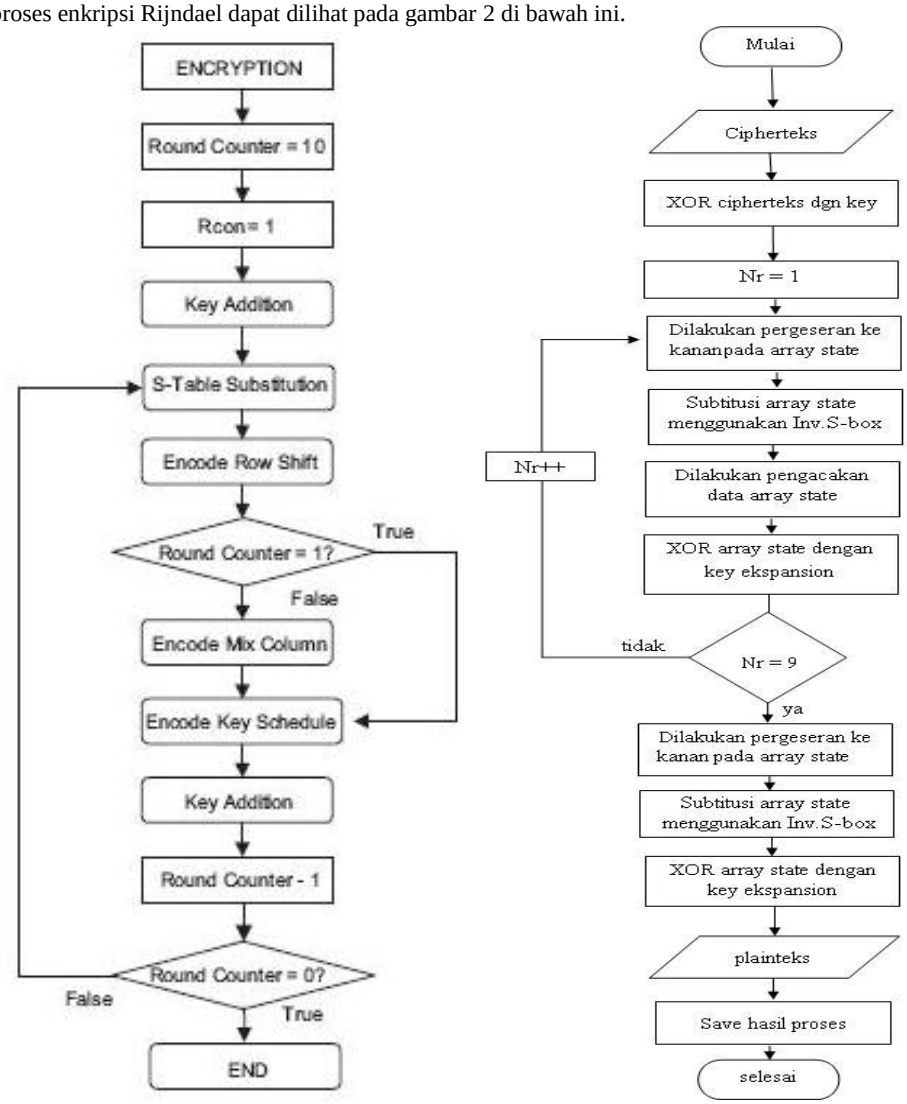
Rumusan masalah yang penulis bahas pada skripsi adalah Hybrid Cryptosistem (Rinjndael & RSA) yang akan diterapkan pada pengamanan file. Kemudian penulis melanjutkannya dengan memberikan beberapa teori tentang topik yang dipilih yaitu Algoritma Rijndael dan Kriptografi RSA. Dilanjutkan dengan penelitian terhadap jurnal-jurnal maupun karya-karya ilmiah yang sudah ada sebelumnya untuk diketahi persamaan dan perbedaannya. Untuk sistem yang dibangun sendiri terdapat 3 fitur yaitu fitur enkripsi, fitur dekripsi dan fitur pembangkitan kunc[8]i.

Untuk fitur enkripsi penulis merancang sistem yang mencakup pembentukan session key rijndael, proses enkripsi rijndael, pembangkitan kunci kriptografi RSA, proses enkripsi RSA dan output pada file. Untuk fitur dekripsi sistem yang dirancang meliputi pembentukan session key rijndael, proses dekripsi rijndael, pembangkitan kunci kriptografi RSA, proses dekripsi RSA dan kemudian

dilanjutkan output pada file. Sistem yang dibangun sendiri berfungsi agar setiap proses enkripsi dan dekripsi pada file berjalan secara otomatis. Kemudian dilanjutkan dengan uji coba pada sistem berbagai session key dan pembangkitan kunci RSA.

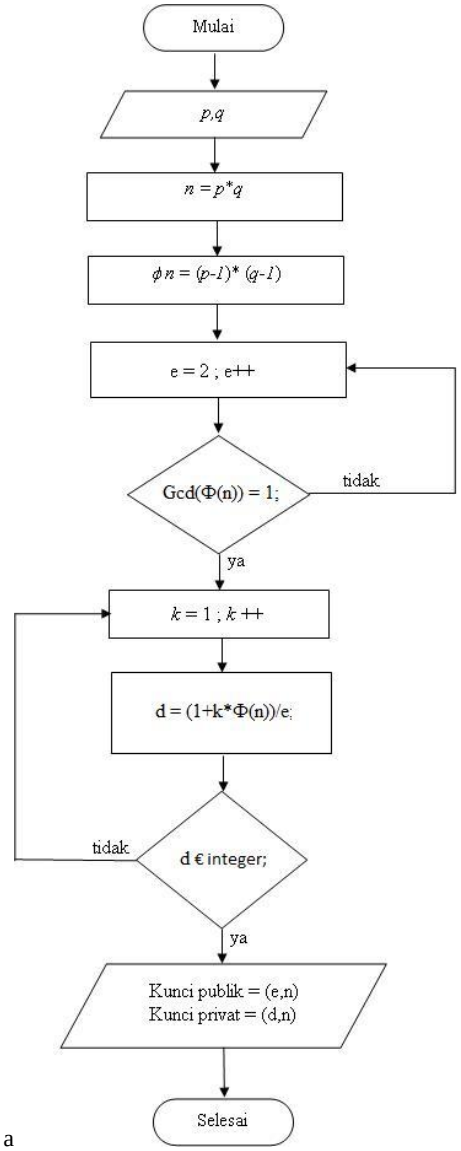
2.2. Flowchart

Flowchart adalah rangkaian bagan yang menggambarkan urutan suatu proses kegiatan dalam mencapai tujuan yang diinginkan. Untuk Flowchart proses enkripsi Rijndael dapat dilihat pada gambar 2 di bawah ini.



Gambar 2. Flowchart Proses Enkripsi dan Dekripsi pada Algoritma Rijndael

Pada gambar 2 dapat dibaca dengan dimulainya proses enkripsi cipherteks maka *round counter* akan mulai dari angka 10 kemudian *Rcon* sama dengan 1 dilanjutkan dengan *key addition.* Lalu dilakukan substitusi dengan s-tabel dengan *encode row shift.* Program kemudian memeriksa apakah *round counter* sudah memiliki nial 1. Apabila belum maka akan dilanjutkan dengan *mengencode Mix Column* dan mengencode *Key Schedule*. Apabila sudah memiliki nilai 1 maka akan langsung dilanjutkan ke proses *encode Key Schedule.* Setelah itu dilanjutkan ke bagian *Key Addition* dan *Round Counter yang mengandung nilai* -1 sehingga program dapat mengecek apakah round counter sudah memiliki nilai 0. Jika belum maka proses enkripsi diulang dari bagian Substitusi S-Tabel. Jika sudah mengandung nilai 0 maka program enkripsi berhasil mengenkripsi file yang dipilih. Proses dekripsi dimulai dengan dimasukkannya cipherteks dan diproses dengan gerbang logika XOR. Kemudian didapatkan nilai Nr=1 lalu dilanjutkan dengan pergeseran ke kanan pada array state, substitusi *array state* menggunakkan inverse S-Box, dilakukan pengacakan *data array state* dan diakhiri dengan mengkonversi *array state* tersebut dengan gerbang logika XOR. Jika nilai Nr yang didapatkan tidak sama dengan 9 maka diulang kembali ke langkah pergeseran array. Jika sudah memiliki nilai 9 maka dilanjutkan dengan melakukan pergeseran ke kanan pada *array state*, subsitusi *array state* menggunakan inv S-Box dilanjutkan dengan konversi dengan gerbang logika XOR. Plainteks akan dihasilkan dan kemudian proses akan disimpan otomatis.[9]



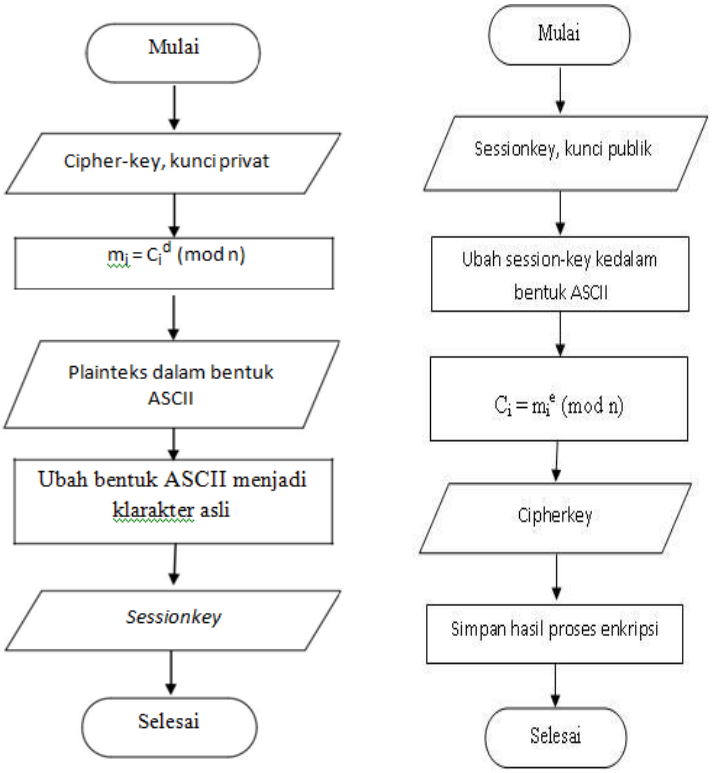
a

Gambar 3. Flowchart Pembangkitan Kunci Publik dan Privat Algoritma RSA

Flowchart pembangkitan kunci privat dan kunci publik pada algoritma RSA dapat dilihat pada Gambar 3.6.
Proses pembangkitan kunci RSA adalah sebagai berikut:

1. Pilih dua buah bilangan prima sembarang p dan q. Jaga kerahasiaan p dan q ini.
2. Hitung n = p * q. Besaran n ini tidak dirahasiakan.
3. Hitung φ(n)= (p-1) * (q-1).
4. Sekali φ(n) telah dihitung, p dan q dapat dihapus untuk mencegah diketahuinya oleh pihak lain.
5. Pilih sebuah bilangan bulat untuk kunci publik e, yang relatif prima terhadap n (GCD(e, φ(n)) = 1) dengan syarat e ≠ (p-1), e ≠ (q-1), dan e < n.
6. Kunci publik (Public Key) = (n,e)
7. Hitung kunci privat (d). Kunci privat dapat dihitung dengan persamaan:
   1 + φ(n)
   dengan syarat e dan d adalah anggota bilangan bulat. Nilai k = 1, 2, 3, ...,n diperoleh nilai d yang bulat. Nilai itu yang akan dipakai sebagai kunci pribadi untuk dekripsi pesan.

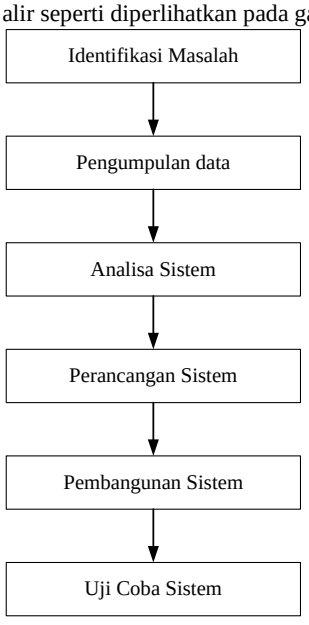
Proses enkripsi Algoritma RSA dijabarkan sebagai berikut :
1. Masukan session key, atau kunci publik
2. Session Key kemudian dikonversi menjadi bentuk ASCII
3. Session Key yang sudah dikonversi akan dienkripsi kembali dengan rumus
ci = mi^e mod n.
4. Dari langkah ketiga akan dihasilkan plainteks dalam bentuk ASCII
5. Ubah kembali bentuk ASCII menjadi karakter asli
6. Akan dihasilkan session key baru yang sudah dienkripsi



Gambar 4. Flowchart Proses Enkripsi Dekripsi Cipherkey pada Algoritma RSA

Flowchart proses dekripsi cipherkey dengan menggunakan kunci privat pada algoritma RSA dapat dilihat pada Gambar 4. Proses dekripsi dapat dijabarkan sebagai berikut :
1. Input cipher key atau kunci privat
2. Cipher key akan diproses dengan rumus mi = ci^d mod n
3. Akan dihasilkan plainteks dalam bentuk ASCII
4. Plainteks tersebut akan diubah dari bentuk ASCII menjadi karakter asli
5. Sessionkey dihasilkan

Langkah-langkah penelitian yang dilakukan adalah identifikasi masalah terlebih dahulu kemudian setelah masalah teridentifikasi dilanjutkan dengan pengumpulan data-data yang berhubungan ataupun membahas mengenai masalah yang diidentifikasi. Dilanjutkan dengan analisa sistem yang berguna untuk mengetahui kelebihan dan kekurangan sistem maupun kegunaan sistem. Kemudian sistem dirancang dan dibangun sesuai dengan identifikasi masalah yang sudah ditemui. Setelah sistem selesai dibangun, dilakukan uji coba sistem yang berguna untuk mengetahui *bug-bug* yang masih ada. Adapun tahapan dan langkah-langkah penelitian skripsi dapat digambarkan dalam bentuk diagram alir seperti diperlihatkan pada gambar 5 di bawah ini.



Gambar 5.. Langkah-Langkah Penelitian

2.3. Lokasi dan Jangka Waktu Penelitian

Penelitian ini dimulai dari bulan Maret 2017 dan akan berlangsung selama 6 bulan. Penelitian ditujukan untuk mengumpulkan data yang diperlukan dalam proses perancangan dan pembuatan sistem. Dalam melakukan penelitian, penulis membuat rencana jadwal agar pelaksanaan kegiatan penelitian dapat berjalan dengan baik. Penelitian dilakukan di kampus dan perpustakaan untuk mengumpulkan data / bahan dan rumah (tempat tinggal) penulis untuk analisis, perancangan, pembangunan dan pengujian. Jadwal penelitian dapat dilihat pada tabel 1. di bawah ini..

Tabel 1.. Rencana Jadwal Pelaksanaan Kegiatan Penelitian

Waktu / Kegiatan	Juni 2017				Juli 2017				Agustus 2017				Sept 2017	
	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Identifikasi Masalah														
Pengumpulan Data														
Analisis Sistem														
Desain / Perancangan														
Pembangunan Sistem														
Pengujian														

2.4. Algoritma AES Rijndael

Pada tahun 1997 NIST (National Institute of Standards and Technology) berniat ingin mencari pengganti algoritma DES, hal ini dikarenakan dalam tempo 96 hari 70 ribu PC berhasil membobol kunci DES, kemudian pada tahun 1998 hanya dalam tempo 22 hari, sehingga dibuat suatu mesin khusus untuk memecahkan algoritma DES. Mesin tersebut terbukti dapat memecahkan 25% kunci DES dalam waktu 2,3 hari dan dapat memecahkan seluruh kunci DES dalam waktu rata-rata 4,3 hari. Karena alasan tersebut NIST ingin mengadakan kompetisi yang diikuti para kriptografer seluruh dunia agar segara mendapatkan pengganti dari algoritma DES[10].

Pada tanggal 9 Agustus 1999, NIST melalui seleksi yang sangat ketat mengumumkan 5 finalis yang akan memasuki seleksi akhir yaitu MARS, RC6, Rijndael, Serpent, dan Twofish. Pada seleksi akhir tanggal 2 Oktober 2000 maka terpilihlah Rijndael sebagai pemenang. Algoritma Rijndael dibuat oleh Dr. Vincent Rijmen dan Dr. Joan Daemen. Evaluasi terhadap algoritma Rijndael tersebut dapat dilihat dari beberapa sudut pandang berikut ini:

1. Dari segi general security
   a. Belum ada serangan yang serius dalam memecahkan skema dari Rijndael
   b. Rijndael menggunakan komponen S-Box non linear.
   c. Rijndael memiliki struktur matematika yang bisa terus mengalami perkembangan.
2. Dari segi enkripsi dekripsi
   a. Memiliki kode yang sangat rumit.
   b. Enkripsi dekripsi Rijndael berbeda satu dengan yang lainnya.
   c. Rijndael mendukung penuh ukuran blok dan ukuran kunci 128 bit, 192bit, dan 256 bit.

Algoritma Rijndael menggunakan substitusi, permutasi, dan sejumlah putaran yang dikenakan pada tiap blok yang akan dienkripsi/dekripsi. Untuk setiap putarannya, Rijndael menggunakan kunci yang berbeda. Algoritma Rijndael mampu menangani panjang kunci dan ukuran blok yang berbeda, panjang kunci dan ukuran blok yang telah ditentukan adalah 128, 192, dan 256.Ukuran blok dapat dihitung dengan (Nb= block length / 32) dan panjang kunci (Nk= key length / 32). Nilai Nb dan Nk yang akan dipakai mempengaruhi jumlah putaran (Nr) yang terjadi pada saat proses enkripsi dan dekripsi. Tabel 2.2 akan menunjukan perbedaan jumlah putaran (Nr) berdasarkan panjang kunci dan ukuran blok yang digunakan. (Mollin, 2007).

Tabel 2. Perbandingan Jumlah Putaran pada Rijndael

Nr	**Nb= 4**	**Nb= 6**	**Nb= 8**
Nk= 4	10	12	14
Nk= 6	12	12	14
Nk= 8	14	14	14

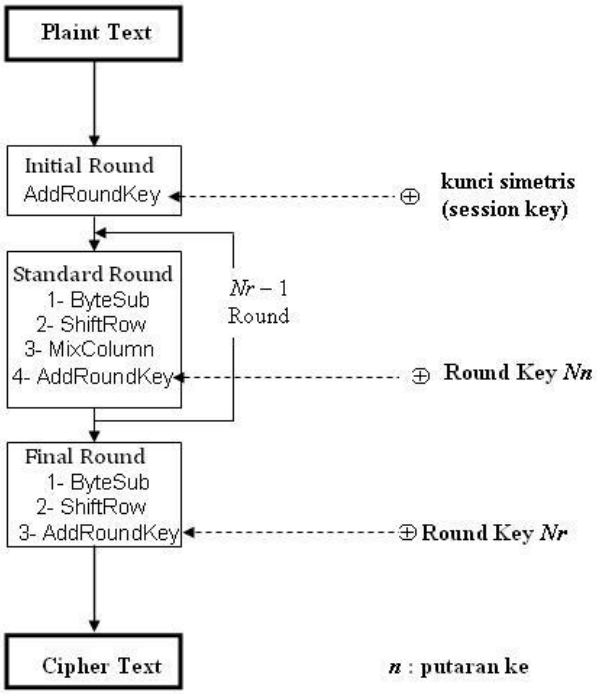
Algoritma Rijndael memiliki tiga para meter yaitu:

1. Plainteks : array yang berukuran 16 byte, yang berisi data masukan.
2. Cipherteks : array yang berukuran 16 byte, yang berisi hasil enkripsi.
3. Key : array yang berukuran 16 byte, yang berisi kunci ciphering.

Algoritma Rijndael beroprasi pada orientasi byte. Tiap elemen dari arraystate diisi dengan 8 bit teks (1 byte) dalam notasi HEX. Urutan pengisian dimulai dari kolom awal (c=0) sampai kolom terakhir (c=3) dan dari baris awal (r=0) sampai baris akhir (s=3). Setiap state pasti mempunyai jumlah baris yang tetap, yaitu 4 baris, sedangkan jumlah kolom tergantung dari besarnya blok data.

2.5. Algoritma Enkripsi AES Rijndael

Proses enkripsi untuk algoritma Rijndael yang beroperasi pada panjang blok 128-bit dengan kunci 128-bit, total putaran (Nr) yang dilakukan hingga diperoleh cipherteks adalah 10 kali putaran. Secara garis besar proses enkripsi algoritma Rijndael diperlihatkan pada Gambar 6 di bawah ini.



Gambar 6. Diagram Proses Enkripsi Rijndael

Urutan proses enkripsi Rijndael dalam mengamankan plainteks yaitu:

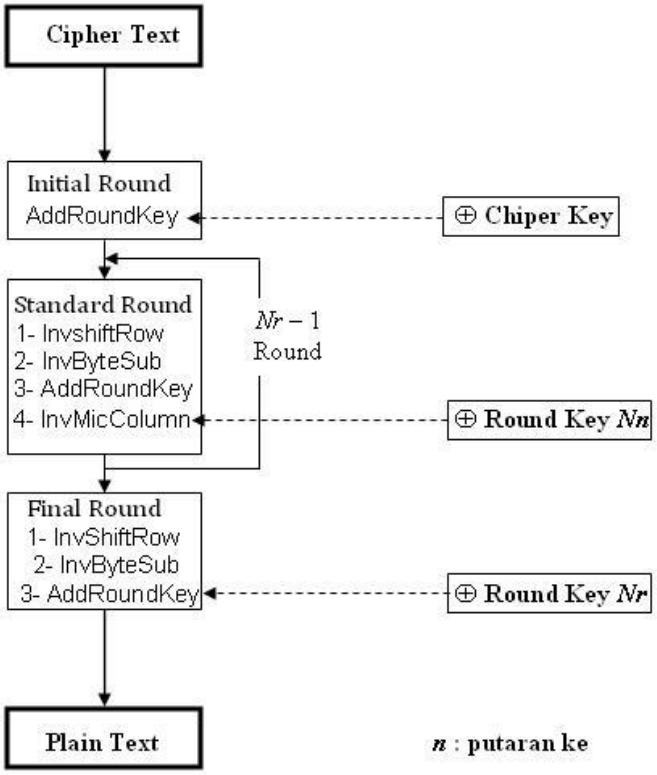
1. Key Expansion
   Pada tahap ini dilakukan ekspansi kunci sesuai dengan panjang kunci dan panjang ukuran blok yang akan digunakan, hasil ekspansi ini disebut dengan Roundkey.
2. Addroundkey
   Untuk proses Addroundkey yang pertama dilakukan XOR antara state awal (plaintext) dengan kunci utama, sedangkan Addroundkey yang selanjutnya pada tiap putaran, merupakan hasil key ekspantion dari kunci utama (sessionkey).
3. Putaran (Nr) sebanyak Nr-1
   Pada proses ini akan dilakukan beberapa putaran, jumlah putaran telah ditentukan seperti yang telah dijelaskan sebelumnya. Pada tahap ini dilakukan 9 kali putaran, yaitu:
   a. Sub Bytes
      Pada proses ini dilakukan subtitusi menggunakan table S-box. Dua digit bilangan HEX yang merupakan representasi 1 byte dari tiap teks menjadi koordinat untuk substitusi pada S-box. Digit pertama sebagai koordinat x dan digit kedua sebagai koordinat y, perpotongan baris x dengan kolom y merupakan nilai yang akan diambil. (Rijmen & Daemen, 2002).
   b. ShiftRows
      Proses pergeseran baris array state dengan menggeser baris ke-r dalam array state ke kiri sebanyak r byte. Baris ke-0 dari blok tidak mengalami pergeseran. Baris ke-1 bergeser 1 byte ke kiri, baris ke-2 bergeser 2 byte ke kiri, sampai akhirnya didapatkan hasil terakhir setelah menggeser baris ke-3 sebanyak 3 byte ke kiri. Berikut adalah table jumlah pergeseran berdasarkan blok Nb. (Rijmen & Daemen,2002).
   c. MixColumns
      Yaitu proses pengacakan data di masing-masing kolom array state. Transformasi MixColumns menggunakan operasi perkalian matriks dengan operasi perkalian dan penjumlahan menggunakan operator pada GF(28) dengan irreducible polynomial (x8 + x4 +x3 + x + 1). Matriks tersebut berdasarkan polinom a(x) mod (x4 + 1) dengan polinom a(x) yang ditetapkan adalah a(x) = 3x3 + x2 + x +2. (Mollin, 2007).
      Transformasi ini dinyatakan sebagai perkalian matriks:
      $s'(x) = a(x) \otimes s(x)$
      $s'0,c = (\{02\} \bullet s0,c) \oplus (\{03\} \bullet s1,c) \oplus s2,c \oplus s3,c$
      $s'1,c = s0,c \oplus (\{02\} \bullet s1,c) \oplus (\{03\} \bullet s2,c) \oplus s3,c$
      $s'2,c = s0,c \oplus s1,c \oplus (\{02\} \bullet s1,c) \oplus (\{03\} \bullet s3,c)$
      $s'3,c = (\{03\} \bullet s0,c) \oplus s0,c \oplus s1,c \oplus (\{02\} \bullet s3,c)$
   d. Addround key
      Dilakukan dengan fungsi XOR antara array state sebelumnya dengan round key. Proses ini akan terus berulang sebanyak Nr-1 sesuai besarnya blok data dan panjang kunci. Setelah semua proses dilakukan sebanyak Nr-1, proses terakhir adalah final round. Pada tahap ini proses MixColumns tidak dilakukan.
4. Final round
   Proses untuk putaran terakhir hanya dilakukan tiga tahap saja, proses dari ketiga tahap tersebut sama seperti proses pada tahap sebelumnya yaitu:
   a. SubBytes.

b. ShiftRows.
c. AddRoundKey.

2.6. Algoritma Dekripsi AES Rijndael

Proses dekripsi algoritma Rijndael tidak jauh berbeda dengan proses enkripsi namun berbeda pada urutan prosesnya saja. Untuk urutan proses dekripsi yaitu:

1. Key Expansion
   Pada proses dekripsi juga dilakukan ekspansi kunci. Kunci pada proses enkripsi kemudian diekspansi terlebih dahulu untuk menghasilkan RoundKey yang akan digunakan pada setiap putaran.
2. AddRoundKey
   Dilakukan proses XOR antara state awal (cipherteks) dengan key terakhir hasil ekspansi. Tahap ini disebut juga initial round.



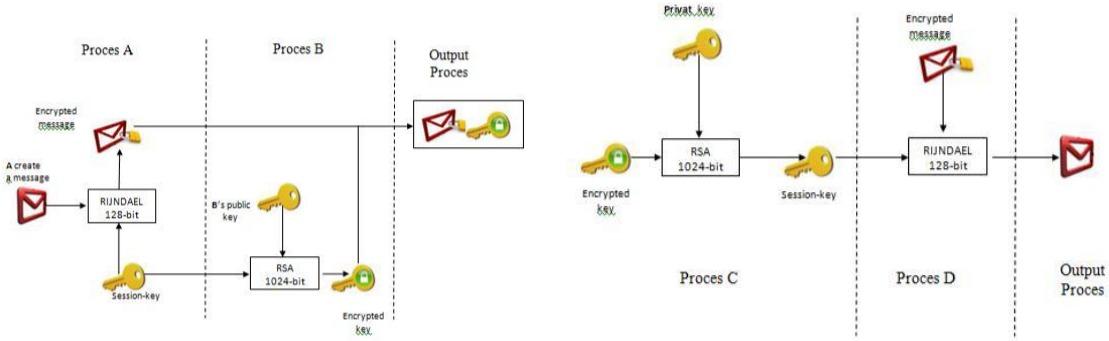
Gambar 7. Diagram Proses Dekripsi Rijndael

3. Putaran sebanyak Nr-1 kali Proses yang dilakukan pada setiap putaran adalah:
   a. InvShiftRow
      Pergeseran baris-baris array state ke kanan dengan aturan pergesaran sama seperti pada tahap enkripsi.
   b. InvByteSub
      Dilakukan substitusi byte dengan menggunakan tabel substitusi kebalikan (inverse S-box).
   c. AddRoundKey
      Pada proses ini dilakukan XOR antara state sekarang dengan round key.
   d. InvMixColumn
      Seperti pada proses MixColumn, InvMixColumn juga dilakukan pengacakan di masing-masing data pada kolom array state.
4. Final round
   Proses untuk putaran terakhir hanya dilakukan tiga tahap saja, proses dari ketiga tahap tersebut sama seperti proses pada tahap sebelumnya, yaitu:
   a. InvShiftRow
   b. InvSubByte
   c. AddRoundKey

2.7. Ekspansi Kunci

Algoritma Rijndael (AES) membuat suatu ekspansi kunci untuk menghasilkan suatu key schedule. Jika ekspansi kunci yang diperlukan Rijndael (AES) Nb(Nr+1) word, sehingga bisa digunakan AES 128 bit, maka 4(10+1)=40 word=44x32 bit=1408 bit subkey. Ekspansi dari 128 menjadi 1408 bit subkey, proses ini disebut dengan key schedule. Subkey ini diperlukan karena setiap round merupakan suatu inisial dari Nb word untuk Nr=0 dan 2 Nb untuk Nr=1, 3 untuk Nr=2,…,11 Nb untuk Nr=10, dari operasi ini akan didapatkan schedule kunci yang berisi array linier 4 byte word (Wi), 0=i(Nr+1). (Arriyus,2008).

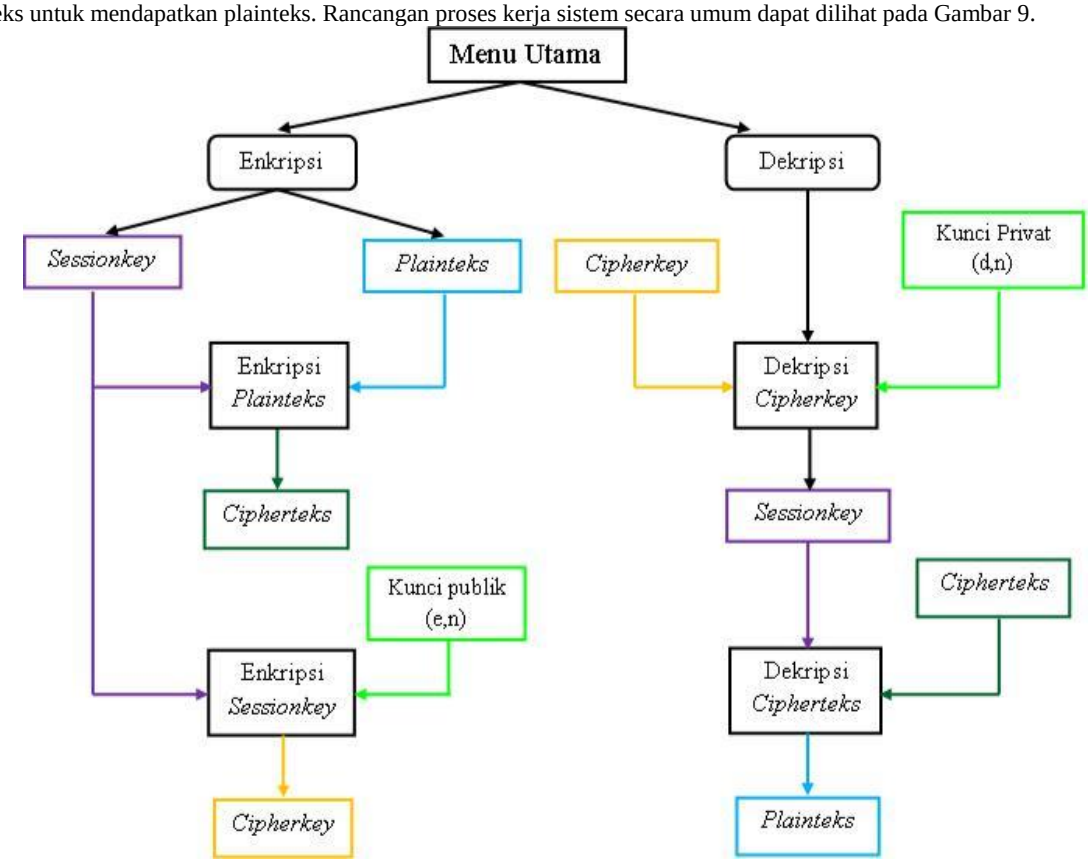
Berikut ini adalah diagram konsep yang akan dilakukan pada tugas akhir ini, diagram konsep ini akan mempermudah dalam perancang sistem yang akan dibuat. Pada tahap awal plainteks terlebih dahulu diproses dengan menggunakan algoritma Rijndael untuk melakukan enkripsi terhadap plainteks kemudian dilanjutkan dengan enkripsi terhadap sessionkey menggunakan algoritma RSA. Hasil akhir dari proses ini akan menghasilkan cipherteks dan cipherkey.



Gambar 8.. Konsep Perancangan Proses Pengamanan Plainteks dan Sessionkey [Sadikin, 2012]

2.8. Rancangan proses kerja sistem secara umum

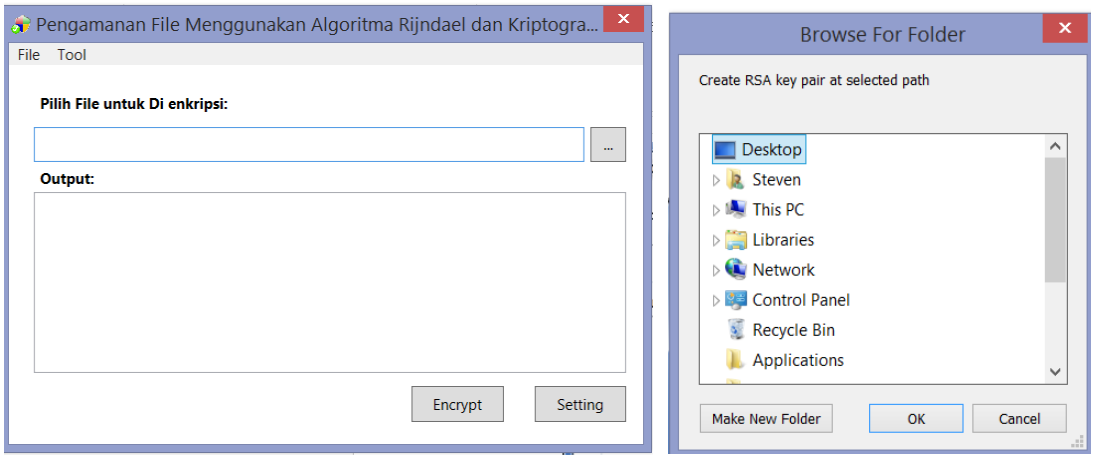
Dalam penerapannya pengirim akan melakukan enkripsi terhadap plainteks dengan menggunakan sessionkey kemudian mengenkripsikan kembali sessionkey tersebut agar tetap terjaga kerahasiaannya dengan menggunakan kunci publik penerima. Untuk proses dekripsi penerima pesan terlebih dahulu mendekripsikan pesan cipherkey yang diterima dengan menggunkan kunci privat yang dimilikinya untuk mendapatkan kembali sessionkey. Setelah sessionkey diperoleh kemudian akan dilakukan dekripsi cipherteks untuk mendapatkan plainteks. Rancangan proses kerja sistem secara umum dapat dilihat pada Gambar 9.



Gambar 9.. Rancangan Proses Kerja Sistem Secara Umum

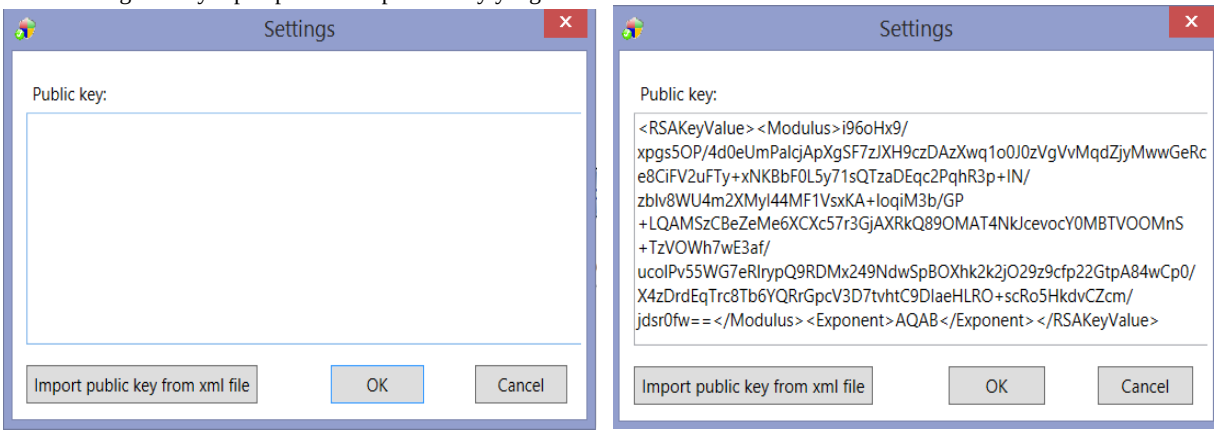
HASIL DAN PEMBAHASAN

Saat aplikasi dijalankan, *form* Utama akan muncul yang mempunyai 2 pilihan menu yaitu file dan tool. Kegunaan dari menu file yaitu untuk mengganti pilihan enkripsi file atau dekripsi file. Sedangkan tool adalah pilihan menu meng-generate public key dan private key.



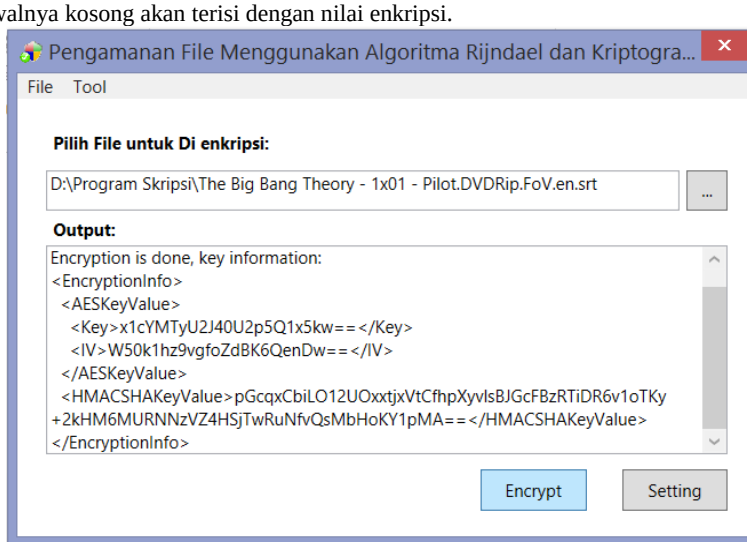
Gambar 10. Tampilan Form Utama dan pilihan Folder

Program enkripsi berjalan secara otomatis saat akan dienkripsi, namun sebelumnya harus menentukan folder dimana file yang akan dienkripsi berada lalu setelah itu merancang public key ataupun private keynya. Sistem akan meminta user untuk mencari folder dimana user ingin menyimpan public dan private key yang akan dibuat.



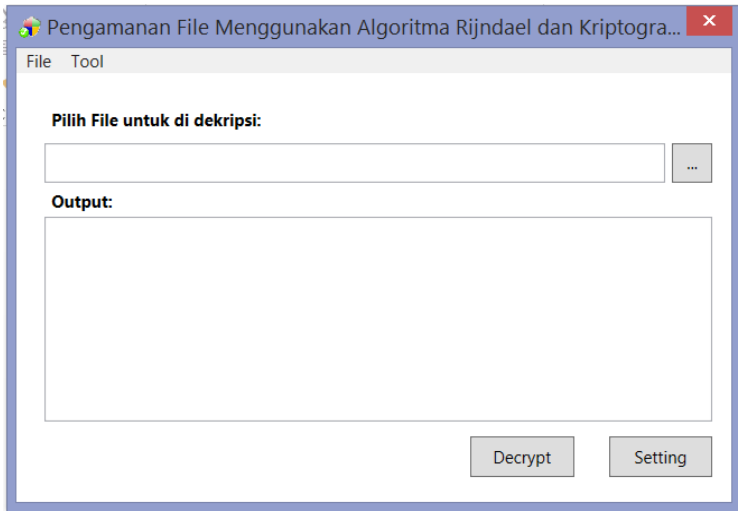
Gambar 11.. Setting Public Key dan Public Key yang telah terisi

Kolom publik key yang awalnya kosong akan terisi dengan nilai enkripsi.



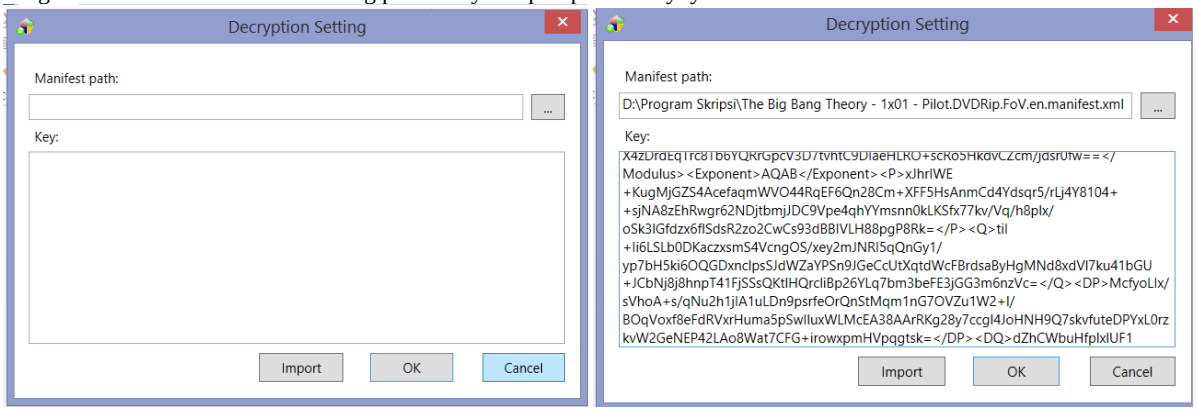
Gambar 12.File yang telah dienkripsi

Hasil terakhir yaitu file yang telah dienkripsi akan masuk ke folder yang telah disediakan secara otomatis.



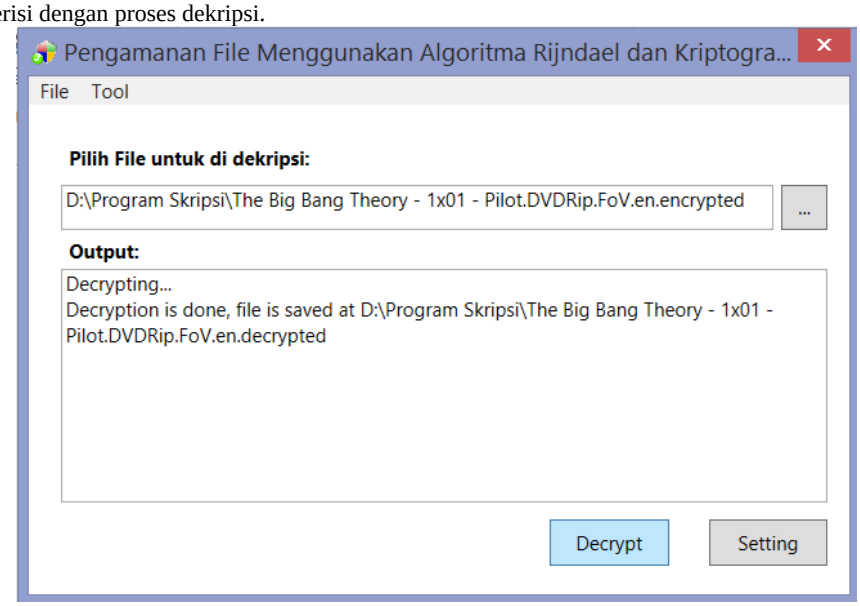
Gambar 13. Tampilan Halaman Dekripsi

Program dekripsi berjalan secara otomatis saat akan didekripsi, namun sebelumnya harus menentukan dimana folder untuk file yang akan digenerate lalu setelah itu merancang public key ataupun private keynya.



Gambar 14. Tampilan Pengisian Kunci Manifest dan Private Key dan Private Key Yang terisi

Kunci manifes terletak di dalam folder yang sama dengan folder yang berisi public dan private key. Nama kunci manifes sama dengan nama file yang ingin dienkripsi namun berbeda ekstensi file. Sesudah memilih kunci manifes dan kunci privat maka kolom yang kosong akan terisi dengan proses dekripsi.



Gambar 15. Hasil Akhir file yang telah didekripsi

Hasil terakhir yaitu file yang telah dienkripsi akan masuk ke folder yang telah disediakan secara otomatis

KESIMPULAN

Setelah menyelesaikan pengembangan aplikasi pengamanan file menggunakan algoritma Rijndael dan kriptografi RSA ini dapat ditarik beberapa kesimpulan yaitu :

1. Sistem yang dibangun mampu mengenkripsi dan dekripsi file yang dipilih saja.
2. Panjang kunci algoritma Rijndael pada sistem ini hanya 128-bit.
3. Hasil penelitian ini diperoleh sistem enkripsi dan dekripsi terhadap plainteks dan kunci simetris (sessionkey) dengan kombinasi algoritma Rijndael dan RSA.
4. Hasil enkripsi plainteks pada sistem yang dibangun berupa kode karakter, sedangkan untuk enkripsi sessionkey berupa kode number.
5. Pengirim dan penerima tidak perlu lagi menyepakati kunci simetris yang digunakan, karena telah terenkripsi oleh algoritma kriptografi asimetris (RSA).

DAFTAR PUSTAKA

[1] R. Munir, "Kriptografi," *Inform. Bandung*, 2006.

[2] R. Munir, "Algoritma & Pemrograman dalam Bahasa Pascal dan C Edisi Revisi," *Andi Yogyakarta*, 2011. [Online]. Available: https://openlibrary.telkomuniversity.ac.id/pustaka/21198/algoritma-pemrograman-dalam-bahasa-pascal-dan-c-edisi-revisi.html. [Accessed: 19-Feb-2020].

[3] T. Limbong, "Pengujian Kriptografi Klasik Caesar Chipper Menggunakan Matlab," *no. Sept.*, vol. 2017, 2015.

[4] N. E. Saragih, "IMPLEMENTASI ALGORITMA ONE TIME PAD PADA PESAN," *J. Ilm. Matrik*, vol. 20, no. 1, pp. 31–40, 2018.

[5] A. M. Hasibuan, "Rancang Bangun Aplikasi Keamanan Data Menggunakan Metode AES Pada Smartphone," *MEANS (Media Inf. Anal. dan Sist.*, vol. 2, no. 1, pp. 29–35, Jun. 2017, doi: 10.17605/JMEANS.V2I1.20.

[6] M. D. Herminingtyas and R. Sholihah, "Implementasi algoritma rc4 untuk enkripsi keamanan data," pp. 1–11, 1987.

[7] Tiepsi and K. Siregar, "Penerapan Algoritma Rijndael untuk Mengamankan Teks," *KAKIFIKOM (Kumpulan Artik. Karya Ilm. Fak. Ilmu Komputer)*, vol. 02, no. 338, pp. 47–53, 2019.

[8] A. Febrianto, "Enkripsi Dan Deskripsi Menggunakan Algoritma RSA," *Fak. Tek. Univ. PGRI Ronggolawe Tuban.*, vol. 53, no. 9, pp. 1689–1699, 2013, doi: 10.1017/CBO9781107415324.004.

[9] J. K. Azhar and S. Yuliany, "Implementasi Algoritma RSA (Rivest, Shamir dan Adleman) untuk Enkripsi dan Dekripsi File .pdf," no. December, 2019.

[10] D. Ariyus, "Kriptografi keamanan data dan komunikasi," *Yogyakarta Graha Ilmu*, 2006.